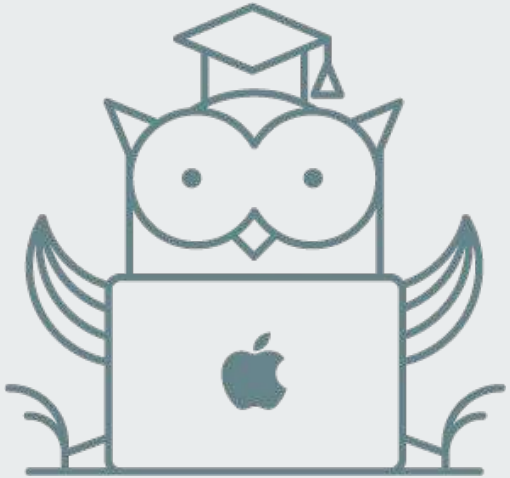
Курс «Администратор Linux»

Сеть наносит ответный удар

Занятие # 9

Дмитрий Молчанов
Григорий Ожегов



Содержание

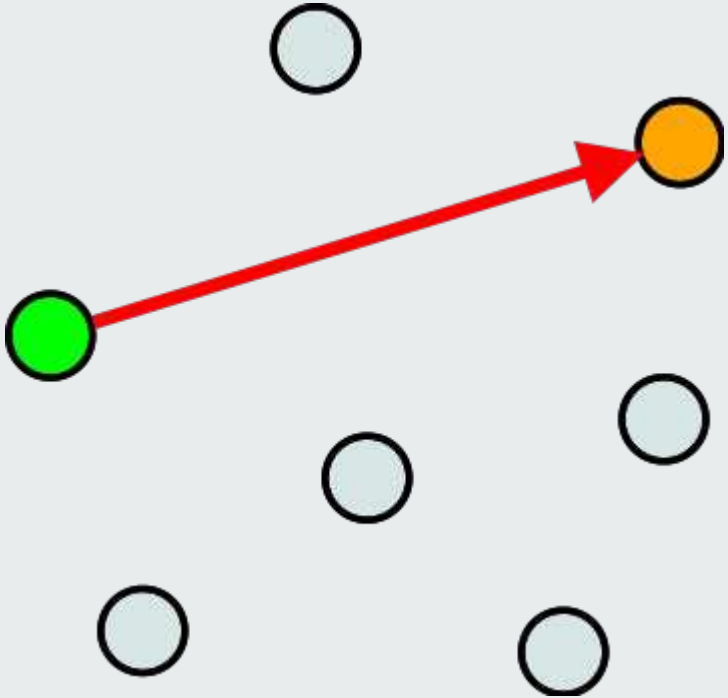
- Типы передачи информации
- Сетевые интерфейсы
  - bond
  - vlan
  - dummy
  - tunnel

Передача данных

- UniCast
- BroadCast
- MultiCast
- AnyCast

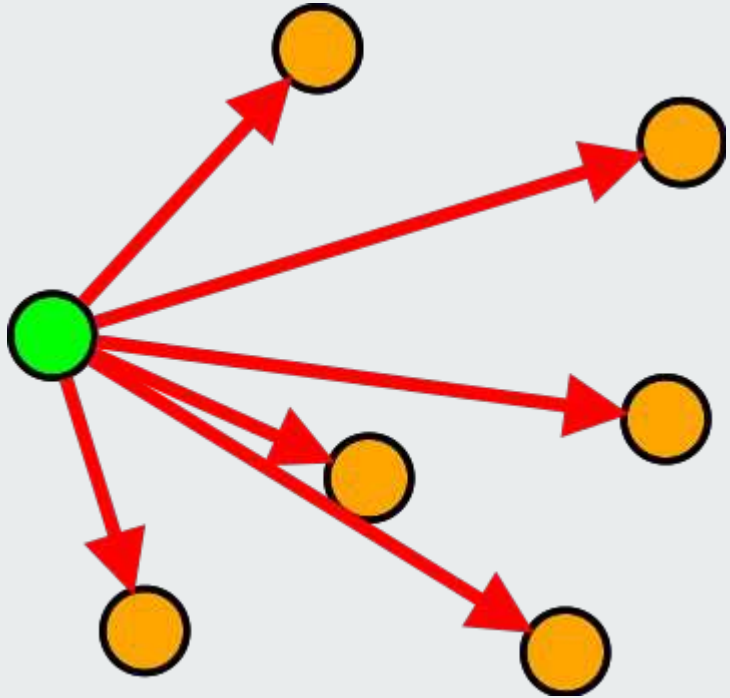
Unicast

Передача peer-to-peer, от одного хоста к одному хосту. Используется в большинстве случаев. Единственный тип передачи поддерживаемый tcp-протоколами.



Broadcast

Широковещательная передача. От одного ко всем. Используется многими протоколами, в частности ARP и DHCP. Широковещательная передача реализована за счет специальных адресов 3го (сетевого) - последний адрес в сети или 255.255.255.255 и 2го (канального) уровня - спецальный mac-адрес FF:FF:FF:FF:FF:FF. При отправке пакета 3го уровня на широковещательный адрес формируется фрейм на широковещательный mac-адрес и обрабатывается сетевым оборудованием.

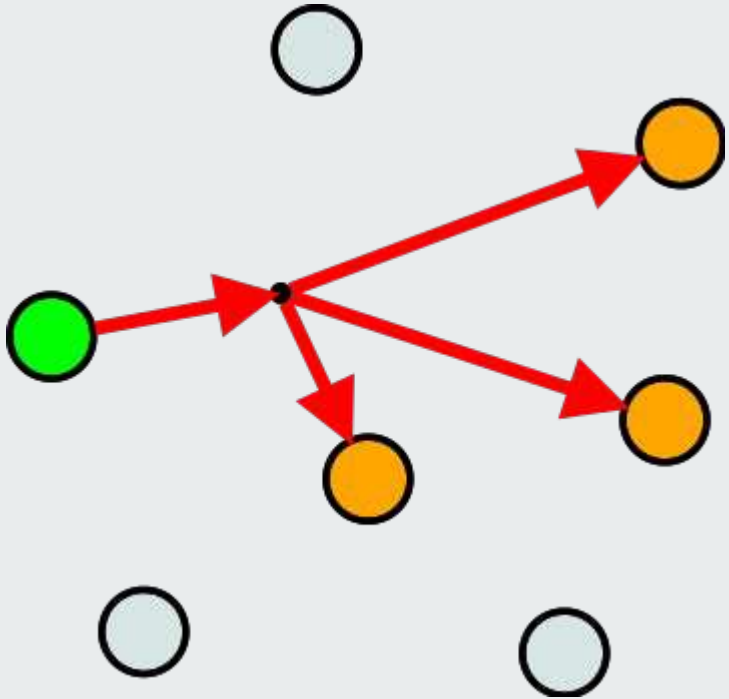


Multicast

Многоадресная передача. От одного ко многим, но не ко всем. Реализована с помощью специальных адресов 2 и 3 уровней и специальны протоколов:

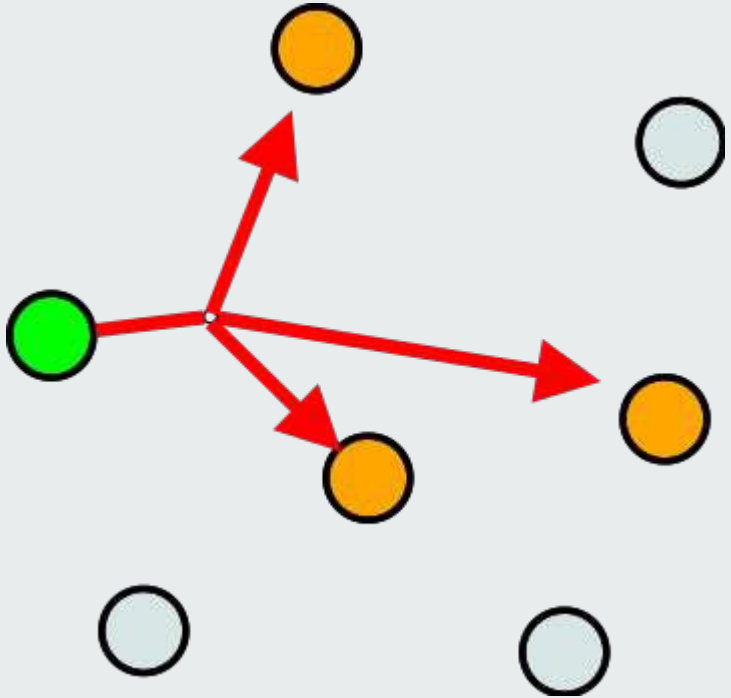
- 224.0.0.0/4 - 3го уровня
- mac-адреса с 1 во 1м(2м) бите 1го октета. маска `0b01000000`.
- IGMP (Internet Group Management Protocol)/MLD (Multicast Listener Discovery)
- PIM-DM/SM, MOSPF, DVMRP - маршрутизация.

Получатели подписываются через IGMP на определенные группы, на сетевом оборудовании эти группы ассоциируются со “специальными” mac-адресами и в итоге трафик получают только те, кто заинтересован в нем.
Не работает с TCP



Anycast

Передача “любому”/”ближайшему” узлу. Технология передачи которая использует особенности маршрутизации (Multipath). За счет особенностей маршрутизации адрес доступен через разные точки и обмен данными идет с ближайшим узлом. Используется для балансировки нагрузки и построения CDN или Георасределенных сервисов.



Многообразие сетевых интерфейсов.

- Bond
- vlan
- Dummy
- Tunnels

Bonding

Объединение сетевых интерфейсов. Есть различные режимы:

#	name	FT	PERF	LB-OUT	LB-IN
0	balance-rr	✓	✓	✓	✗
1	active/backup	✓	✗	✗	✗
2	balance-xor	✓	✓	✓	✗
3	broadcast	✓	✗	✗	✗
4	802.3ad	✓	✓	✓	✓
5	balance-tlb	✓	✓	✓	✗
6	balance-alb	✓	✓	✓	✓?

Bonding: приготовление, mode1

1. Создается виртуальный мастер-интерфейс bond*N* (ifcfg-bondN):

```
= ifcfg-bond0 =
DEVICE="bond0"
ONBOOT=yes
TYPE=Bond
BONDING_MASTER=yes
IPADDR=192.168.10.1
PREFIX=24
BOOTPROTO=static
BONDING_OPTS="mode=1
miimon=100"
==============
```

на этом интерфейсе конфигурируется IP

2. Конфигурируются слейв-интерфейсы:

```
= ifcfg-enp0s9 =
DEVICE=enp0s9
ONBOOT=yes
SLAVE=yes
MASTER=bond0
BOOTPROTO=none
================

= ifcfg-enp0s10 =
DEVICE=enp0s10
ONBOOT=yes
SLAVE=yes
MASTER=bond0
BOOTPROTO=none
=================
```

3.
```
# cat /proc/net/bonding/bond0
Ethernet Channel Bonding Driver: v3.7.1 (April 27, 2011)

Bonding Mode: fault-tolerance (active-backup)
Primary Slave: None
Currently Active Slave: enp0s9
MII Status: up
MII Polling Interval (ms): 100
Up Delay (ms): 0
Down Delay (ms): 0

Slave Interface: enp0s9
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:0c:e6:67
Slave queue ID: 0

Slave Interface: enp0s10
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:6d:b1:d6
Slave queue ID: 0
```

Bonding: приготовление, mode4

mode 4 - требует поддержки со стороны свича, но в остальном не сильно отличается от mode 1

VLAN

Перед администраторами linux и сетей очень часто возникает задача изоляции сегментов друг от друга. Их можно решить физическим разделением оборудования, но это не всегда приемлемо, т.к. количество портов ограничено, а стоимость оборудования не нулевая (и карт и свичей). На помощь приходят VLAN - они позволяют виртуализировать логические сети на одном оборудовании, как гипервизор позволяет запускать несколько виртуальных машин на одном компьютере.

Существует несколько подходов к организации vlan на L2:

1. 802.1q (Vlan-tagging)
2. Port grouping

VLAN:802.1q

Стандарт 802.1q приносит нам понятие режима порта:

- native/access
- trunk/tagged
- mixed
- native vlan

Обычно сервера подключаются в access-режиме и тогда все прозрачно для администратора, но это дает возможности пользоваться vlan'ами. Полностью возможности технологии раскрываются, когда порт переводтися в trunk/tagged режим и появляется возможность получать несколько VLAN'ов по одному порту за счет добавления vlan-тега. Это требует дополнительной конфигурации, как со стороны оборудования, так и со стороны операционнной системы.

VLAN

Заголовок 802.1q размером 32 бита добавляется внутрь заголовка ethernet фрейма и увеличивает размер фрейма на 4 октета (32 бит), важно чтобы сетевое оборудование понимало это. Под идентификатор vlan'а выделено 12 бит (max 4096).

При посылке через vlan-интерфейс и передаче в родительский интерфейс к фрейму добавляется 802.1q заголовок и контрольная сумма фрейма пересчитывается. Это L2 информация, поэтому 802.1q это технология 2го уровня и не маршрутизируется.

При приеме фрейма на интерфейс система смотрит наличие у него 802.1q заголовка и, если нету, отправляет его на физический интерфейс (native vlan), а если есть - в соответствующий vlan-интерфейс.

VLAN для L3 называется IP-VPN и реализуется через технологию MPLS.

VLAN: Приготоволение.

- Убедиться, что vconfig установлен:
```
[root@Net-R1 network-scripts]# rpm -q vconfig
vconfig-1.9-16.el7.x86_64
```
- Конфигурация parent-интерфейса:
```
[root@Net-R0 network-scripts]# cat ifcfg-enp0s10
ONBOOT=yes
BOOTPROTO=none
DEVICE=enp0s10
NM_CONTROLLED=no
```
- Конфигурация vlan-интерфейса:
```
[root@Net-R0 network-scripts]# cat ifcfg-enp0s10.2
ONBOOT=yes
VLAN=yes
BOOTPROTO=static
TYPE=Ethernet
DEVICE=enp0s10.2
NM_CONTROLLED=no
IPADDR=172.16.2.1
PREFIX=24
```

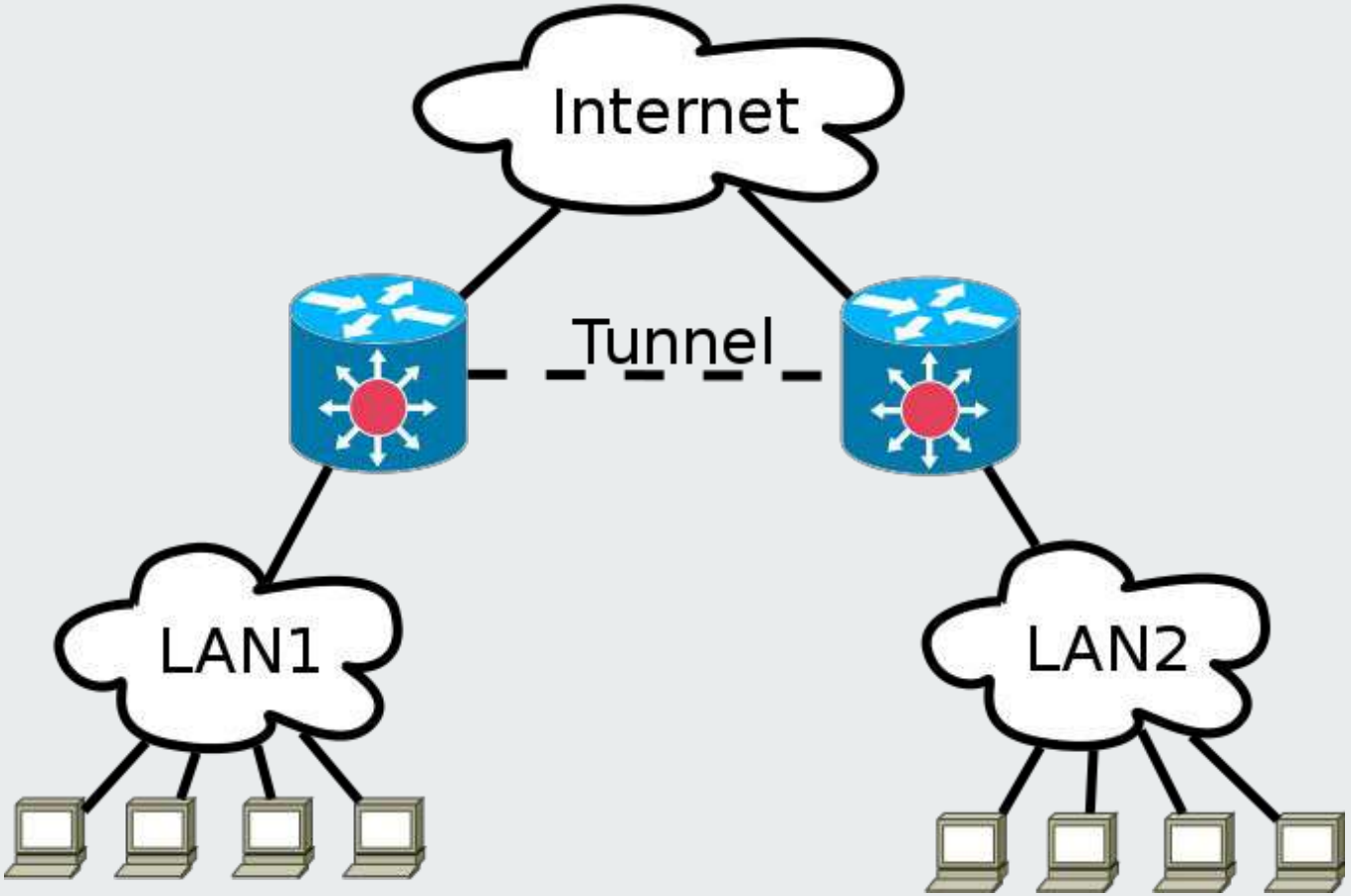
Профит:

```
[root@Net-R0 network-scripts]# ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT
qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode
DEFAULT qlen 1000
    link/ether 08:00:27:4e:67:91 brd ff:ff:ff:ff:ff:ff
3: enp0s8: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master
bond0 state UP mode DEFAULT qlen 1000
    link/ether 08:00:27:8b:da:6c brd ff:ff:ff:ff:ff:ff
4: enp0s9: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master
bond0 state UP mode DEFAULT qlen 1000
    link/ether 08:00:27:b2:c7:6a brd ff:ff:ff:ff:ff:ff
5: enp0s10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
mode DEFAULT qlen 1000
    link/ether 08:00:27:0d:43:6d brd ff:ff:ff:ff:ff:ff
7: enp0s10.2@enp0s10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UP mode DEFAULT qlen 1000
    link/ether 08:00:27:0d:43:6d brd ff:ff:ff:ff:ff:ff
```

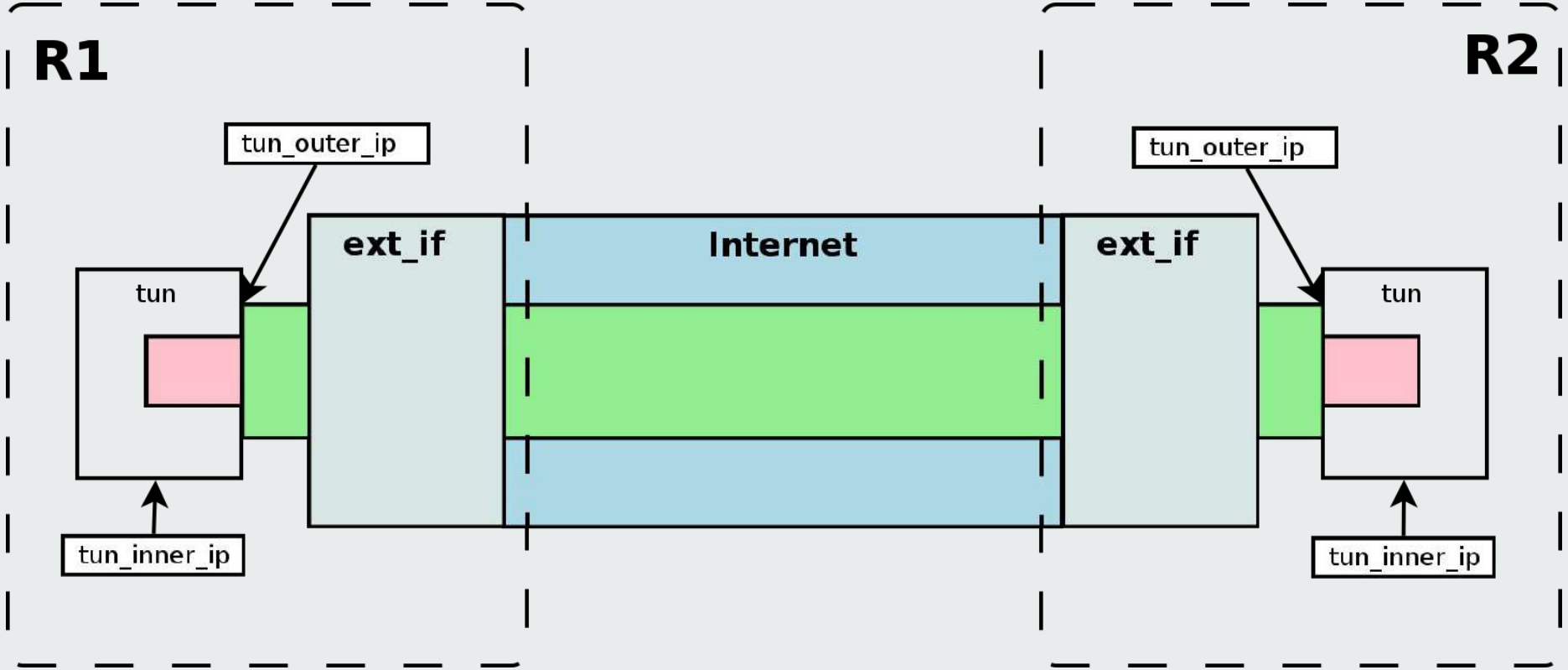
Tunnels

Для объединения частных сетей через интернет есть возможность туннелирования - создания транспортных потоков в которых l3-пакеты частных сетей инкапсулируются в пакеты публичных сетей. Для организации туннелей канального используются разные протоколы, чаще всего можно встретить gre (Generic Router Encapsulation, ip/47) и ipip (IPENCAP ip/4) туннелирование . Также туннелирование возможно без реализации интерфейса канального уровня, например ipsec в туннельном режиме, но подходит только для ip. IP-туннели это point-to-point интерфейсы.

Применение туннелей



Tunnels+l2


R1
R2
tun_outer_ip
tun_outer_ip
ext_if
Internet
ext_if
tun
tun
tun_inner_ip
tun_inner_ip

Tunnels приготовление ручками

```
[root@Net-R0 network-scripts]# ip tun add gre2 mode gre remote 192.168.1.126  local 192.168.1.1  ttl inherit
[root@Net-R0 network-scripts]# ip add add 192.168.55.1 peer 192.168.55.2/32 dev gre2
[root@Net-R0 network-scripts]# ip link set gre2 up
[root@Net-R0 network-scripts]# ip addr show dev gre2
21: gre2@NONE: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1476 qdisc noqueue state UNKNOWN qlen 1
    link/gre 192.168.1.1 peer 192.168.1.126
    inet 192.168.55.1 peer 192.168.55.2/32 scope global gre2
       valid_lft forever preferred_lft forever
    inet6 fe80::5efe:c0a8:101/64 scope link
       valid_lft forever preferred_lft forever
```

Tunnels: приготовление

Side A

```
[root@Net-R0 network-scripts]# cat ifcfg-gre1
DEVICE=gre1
NAME=gre1
NM_CONTROLLED=yes
BOOTPROTO=none
TYPE=GRE
ONBOOT=yes
MY_OUTER_IPADDR=192.168.1.1
PEER_OUTER_IPADDR=192.168.1.129
MY_INNER_IPADDR=192.168.100.1
PEER_INNER_IPADDR=192.168.100.2
[root@Net-R0 network-scripts]# ifup gre1
```

Side B

```
[root@Net-R2 network-scripts]# cat ifcfg-gre1
DEVICE=gre1
NAME=gre1
NM_CONTROLLED=yes
BOOTPROTO=none
TYPE=GRE
ONBOOT=yes
MY_OUTER_IPADDR=192.168.1.129
PEER_OUTER_IPADDR=192.168.1.1
MY_INNER_IPADDR=192.168.100.2
PEER_INNER_IPADDR=192.168.100.1
[root@Net-R2 network-scripts]# ifup gre1
```

dummy interface

Иногда возникает необходимость быстро добавлять-удалять адреса в системе.
Возможные пути решения:

- ip addr add/del - ручная работа, можно ошибиться
- interface aliases (enp0s3:1) - привязка к физическому/родительскому интерфейсу. Как вариант можно использовать алиасы на lo, но это как-то нелогично.
- **dummy interface** - интерфейс пустышка, не привязан ни к какому физическому интерфейсу у каждого адреса/адресов свой интерфейс. Позволяет заводить интерфейсы “per service” и управлять отдельно всей адресацией относящейся к сервису.

В основном в этом возникает необходимость в “продвинутых” сетапах с динамической маршрутизацией/vrrp

dummy interface: приготовление

1. Конфигурирование модуля

```
[root@linux-demo network-scripts]# cat /etc/modprobe.d/dummy.conf
options dummy numdummies=4
[root@linux-demo network-scripts]# modprobe dummy
```

2. Конфигурирование интерфейса (no NetworkManager)

```
ONBOOT=yes
BOOTPROTO="static"
DEVICE=dummy0
NM_CONTROLLED=no
IPADDR=192.168.44.1
PREFIX=32
```

3. Активация интерфейса

```
ifup dummy0
```

Спасибо за внимание

Дмитрий Молчанов
Григорий Ожегов